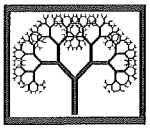
Paper 106



Proceedings of the Tenth International Conference on Civil, Structural and Environmental Engineering Computing,
B.H.V. Topping (Editor),
Civil-Comp Press, Stirling, Scotland.

Combining a Similar Coefficient Identification Algorithm with the Boundary Element Method

L. Hermanns†, I. del Rey‡, A. Fraile‡ and E. Alarcón‡
†Centre for Modelling in Mechanical Engineering (F2I2-CEMIM)
‡Department of Structural Mechanics and Industrial Constructions
Polytechnical University of Madrid, Spain

Abstract

This paper shows how the combination of a similar coefficient identification algorithm (SCIA) with the boundary element method (BEM) permits to treat problems which may or may not present any type of global symmetry and have more than 35,000 degrees of freedom (dofs) using a conventional personal computer. The results obtained with an implementation for dynamic problems in viscoelastic media solved in the frequency domain for a typical soil structure interaction problem (SSI) are presented. This particular problem is just one example for applications where a considerable part of the discretized domain may be meshed with equally sized elements. Many entries in the coefficient matrix resulting from these zones are therefore repeatedly calculated and stored. The benefits of the application of the SCIA are twofold. First, the identical coefficients are only calculated once, and second, for those problems in which the coefficient matrix does not fit into the computer's RAM, still all different coefficients may be stored and the matrix vector products in the GMRES algorithm modified to recover all coefficients from the stored ones by simple operations like changing the local node numbering or rotations.

Keywords: similar coefficient identification algorithm (SCIA), boundary element method (BEM), wave propagation, soil-structure interaction (SSI), pile foundations, dynamic stiffness.

1 Introduction

During the last decades the BEM has often been applied to solve SSI problems of small to moderate size as it does not suffer from certain problems like reflections at discretization boundaries which the domain type approaches like the Finite Element Method usually have to deal with. Apart from that, the number of dofs in the case of 3-d models may reach several hundreds of thousands which clearly asks for

powerful hardware equipment and even then leads to very long computation times. Even when employing boundary elements the number of dofs reaches several ten thousands.

As a consequence of the number of dofs dealt with, usually iterative methods are employed to solve the resulting system of equations. When using the collocation method in combination with the conventional BE formulation, the resulting coefficient matrix is non symmetric and fully populated for every block representing a subregion of the model, i.e. a zone of constant material properties. An iterative method widely used in this context is the GMRES algorithm [5]. During the solution process a matrix vector product has to be calculated wherefore the coefficient matrix has to be provided. When dealing with Finite Elements the computational cost for calculating its entries is not very high so that, in the case of very big problems, they can be calculated in every iteration step anew, thereby avoiding memory problems. While in general this approach may also be used in boundary element implementations in practice it is not as the computational cost for the generation of the coefficients is considerably higher than for Finite Elements leading to excessive computation times. This indicates a disadvantage of the BEM, in comparison with the FEM, for problems having a number of dofs which leads to a coefficient matrix that can not be stored in the RAM of one computer. One solution to this problem is obvious, the distribution of the work among several computers (domain decomposition, distributed computing), nevertheless, this approach requires important computational resources to be available which may not always be the case.

Approaches like the Fast Multipole Method (FMM) result in considerable reductions in computation time however, the price to pay for the approximations introduced are results of less accuracy.

In the present paper a somehow different approach is described which can be combined with domain decomposition in order to increase its efficiency; however, it may also be used directly with a conventional boundary element implementation. Computational grids used when dealing with SSI problems contain regular zones like plain soil surfaces as well as irregular ones like arbitrarily shaped footings. The part of the coefficient matrix resulting from the regular zones, when meshed with equally sized elements, presents a high number of repetitive entries as there are many pairs of collocation points and elements that have the same geometrical configuration, i.e. one may identify certain patterns in the mesh. The proposed approach consists in identifying the pairs of collocation points and elements which result in the same coefficients and, in a second step, reorganizing the coefficient matrix calculation as well as the solution algorithm, to take advantage of the gathered information.

The remaining part of the paper is organised as follows: In the following section the field equations are briefly described while in section 3 some details of the developed computer program are given. In section 4 the similar identification

algorithm is described in more detail. Section 5 contains some numerical results and a comparison with those obtained using an axisymmetric formulation before some conclusions are given in section 6.

2 Field equations in the frequency domain

Combining the stress equilibrium equations with kinematic relations for small strains and supposing a linear elastic material governed by Hooke's law, the Navier equations (1) are obtained, which govern the motion of homogeneous, isotropic, linear elastic bodies.

$$\tilde{\mu}u_{i,jj} + (\tilde{\lambda} + \tilde{\mu})u_{j,ji} + \rho b_i = \rho \ddot{u}_i \ , \tag{1}$$

where u_i are the displacements, $\tilde{\lambda}$ and $\tilde{\mu}$ are the Lamé parameters, b_i represents a body force per unit mass and ρ denotes the material's density. A proper set of boundary and initial conditions closes the mathematical description of the problem. When dealing with time harmonic problems with frequency ω one arrives at the following expression.

$$\tilde{\mu}u_{i,jj} + (\tilde{\lambda} + \tilde{\mu})u_{j,ji} + \rho b_i = -\omega^2 \rho u_i \tag{2}$$

The study of time harmonic problems is of great interest as solutions to transient problems may be found by superposition and the application of Fourier or Laplace transforms. Applying the correspondence principle to equation (2), i.e. using complex valued Lamé parameters, the motion of homogeneous, isotropic, linear viscoelastic bodies may be treated as well. The relations between the real and complex valued Lamé parameters are as follow:

$$\lambda = (1 + 2\beta_\lambda \sqrt{-1})\tilde{\lambda} \ , \tag{3a}$$

$$\mu = (1 + 2\beta_\mu \sqrt{-1})\tilde{\mu} \ , \tag{3b}$$

where β is a parameter, often called damping factor, which may depend or not on the frequency ω modelling thereby viscous or hysteretic damping respectively. In general different β may be used for λ and μ as indicated in (3) by the different sub-indices however in the present study $\beta_\lambda = \beta_\mu$ is used leading to following equation used as starting point for the boundary element method.

$$\mu u_{i,jj} + (\lambda + \mu)u_{j,ji} + \rho b_i = -\omega^2 \rho u_i \tag{4}$$

3 Description of the BEM implementation

In the present section the steps necessary to arrive at a boundary element formulation for the problem under study are only briefly described, a more detailed description may be found in standard text books on BEM like [1], [2] or [3]. Apart from that, some implementation details are given.

Using a weighted residual approach, and supposing that the actual as well as the weighting field are harmonic, the following boundary element formulation can be obtained [2].

$$c_{ij}^k u_i^k + \int_\Gamma p_{ij}^* u_i d\Gamma = \int_\Gamma u_{ij}^* p_i d\Gamma + \int_\Omega u_{ij}^* \rho b_i d\Omega, \tag{5}$$

where Ω is the domain of study bounded by Γ, c_{ij}^k depends on the local geometry at the collocation point including the free term which arises in the limit process when the expression valid for points inside the domain Ω is taken to the boundary, p_i denotes the traction component in direction i while u_{ij}^* and p_{ij}^* are the displacement and traction kernel respectively. The kernel functions may be found in [2]. Supposing that the body forces b_i vanish and that the boundary Γ is divided into M elements, the integrals in (5) may be expressed in terms of a summation over all elements leading to (6).

$$c_{ij}^k u_i^k + \sum_{m=1}^{M} \int_{\Gamma_m} p_{ij}^* u_i d\Gamma = \sum_{m=1}^{M} \int_{\Gamma_m} u_{ij}^* p_i d\Gamma \tag{6}$$

In Figure 1 the implemented incompatible nine node element is presented, whose shape functions are given in the Appendix. The circles indicate the position of the nodes used for the geometrical interpolation while the rectangles indicate the positions of the support nodes for the interpolation of the field variables, i.e. the displacement and the traction.

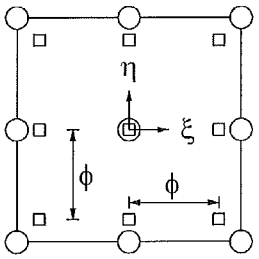


Figure 1: incompatible nine node element

As the surface is smooth at the collocation points, the c_{ij}^k coefficients in equation (6) take the simple form $0.5\ \delta_{ij}$ where δ_{ij} is the Kronecker delta. The position of the supporting nodes for the variable interpolation is an input datum of the program and is defined in natural coordinates (ξ and η) by means of a unique value ϕ between zero and one. This type of element has been chosen for two reasons. As one of the main fields of application is the determination of the dynamic stiffness of surface, partly embedded and pile foundations, this element does not suffer from the corner problem as no collocation point is located at any corner. The second reason has to do with multi region problems. Incompatible elements share their degrees of freedom only with one element of a neighbour region, which results in a small number of different cases one has to deal with, i.e. an interface element may not be at the same time an element where boundary conditions are imposed. The disadvantage of the chosen element lies in the number of dofs which is considerably higher compared to an implementation using compatible elements of the same order.

Describing the geometry as well as the variation of the displacements and tractions over an element by means of shape functions and combining the coefficients resulting from the integrals containing the traction kernel p_{ij}^* with the first summand of expression (6) , the following system of equations is obtained.

$$\underline{H}\ \underline{U} = \underline{G}\ \underline{P} \tag{7}$$

Incorporating the boundary conditions in the displacement and tractions vectors ($\underline{U}$ and $\underline{P}$) and collecting all unkowns in a vector $\underline{X}$ one arrives at the following system of linear equations in complex variables.

$$\underline{A}\ \underline{X} = \underline{F} \tag{8}$$

In the following some features of the implementation are listed.

- three different node numbering schemes are supported (Normal, Gambit, Ansys)
- adjustable collocation point position defined by input datum
- automatic identification of interface elements
- identification of similar collocation point / element pairs (described in the next section)
- direct method for the integration of Cauchy Principal Value integrals [4]
- adaptive integration technique by successive subdivision of the integration domain for non-singular integrals
- solution of the system of equations (8) by LU decomposition or GMRES depending on the number of dofs
- automatic detection of available RAM and adaptation of the solution strategy

When dealing with 3-d problems, the generation of suitable meshes may take a lot of time. Therefore the node numbering schemes according to two commercial preprocessors (Ansys and Gambit) have been implemented.

Depending on the available memory the coefficient matrix is completely set up or the entries are repetitively generated from the stored key coefficients for the matrix vector product in the GMRES algorithm. The program is coded in Fortran 90 and uses double precision variables.

4 Similar coefficient identification algorithm (SCIA)

First of all the meaning of the word similar in the sections title deserves an explanation. In the following the word pair is used to describe a combination of a collocation point and an element that is integrated over. Similar coefficients are those resulting from pairs that are located in subregions with identical material properties and have the same or a similar geometrical configuration. With similar configuration is meant that one pair may be transformed into another by translation, rotation, changing its local node numbering scheme or a combination of these actions. In the present version only multiples of 90 degrees rotation angles are supported.

In a first step groups of pairs are identified which share the distance of the pairs calculated between the collocation point and the central node of the element that is integrated over, having the same element size and being located in subregions with identical material properties.

For each group an identification matrix is set up, i.e. the matrix entry in line 3 column 6 will describe the relationship between the third and the sixth pair of that group by means of an integer value which permits to identify the axes of rotation, the rotation angle and the local node numbering scheme. To this end the members of each group are compared in more detail, i.e it is checked whether a simple translation is sufficient or if additional transformations like a change of the local node numbering scheme is necessary to convert the first pair into the second. This is accomplished by simple loops. Usually the identification process is interrupted after having found three or four similar pairs for each one as it does not make sense to compare every possible combination of pairs in the group, i.e. calculate all matrix entries. The number of similarities found before the search is interrupted is an input datum.

After having identified the similarities within a group, a list is set up with an optimized calculation order to minimize the actual integrations to be performed. For each pair a unique integer value is stored in the list which permits to recover the element number and the position of the collocation point. Apart from that, the position of the source element in the list for the transformation and the identification code previously determined is stored for every pair, which permits to obtain the resulting coefficient matrix entries for the pair in turn. If the code stored is zero, the

corresponding integral has to be calculated by means of explicit numerical integration, as it cannot be obtained by transformation of yet calculated matrix entries. Concatenating the lists of all groups one gets a calculation guide for all entries of the coefficient matrix. After that, the boundary conditions for every pair's element are compared with those corresponding to elements of pairs which are calculated by transformation of the former one. If the boundary conditions of all these elements are of the same type and zero, then only one of the element matrices ($\underline{g}$ and $\underline{h}$) has to be calculated and stored.

The time employed to identify similar coefficients is recovered as the number of explicit integrations is greatly reduced. The application of the SCIA permits to exploit symmetric parts, if present, of models which lack of global symmetry planes. Sometimes it may result in less computation time if a model is subdivided in smaller, equally shaped subregions.

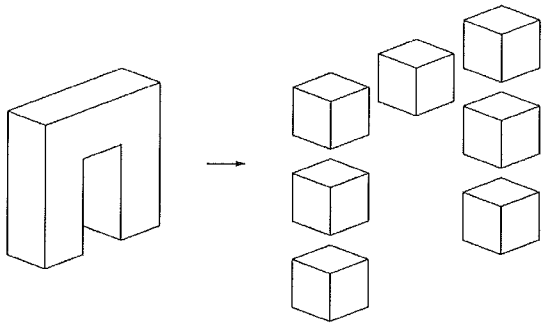


Figure 2: subdivision of a region in several smaller ones of the same shape

In Figure 2 an example is pictured where a type of massive portal has been subdivided into cubes. The SCIA detects the similarities between the pairs of all cubes, what results in very few explicit integrations at the cost of increasing artificially the number of dofs by introducing interface elements between the different cubes. Experience gained so far with the SCIA indicates that the total computation time is greatly reduced although the number of dofs has been increased.

The case where a model of one pile is extended in order to model a pile group of identical piles may serve as another example, as the SCIA will detect the similarities between the piles thus leading to reduced memory requirements and calculation time.

5 Numerical results

In order to test the performance of the described implementation, a typical SSI problem has been chosen: the vertical dynamic stiffness of a single pile foundation

in layered soil. The geometrical and material data are presented in Figure 3 while Figure 4 shows an isometric view of half of the mesh used.

A single pile model is interesting as the results may be compared to those obtained with an implementation that takes advantage of the axial symmetry. To this end a computer program has been coded using a three node incompatible element and following the approach discussed in [2], i.e. using the full 3-d fundamental solution and circular elements.

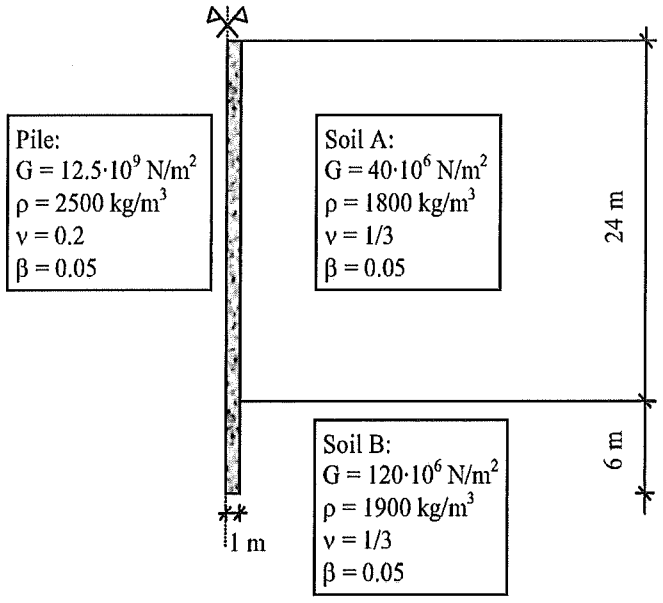


Figure 3: geometrical and mechanical properties of the pile model

Several mesh densities have been tested for the axisymmetric model to minimize the influence of the mesh on the results. Note that both models have similar extensions in the direction perpendicular to the pile axis. In the case of the axisymmetric model the radius of the surface disc is 30 m.

As may be observed in Figure 4 there are transition zones where the mesh quality is not very good however, as can be see in Figure 5 the results obtained are quite satisfactory even with the mesh employed. The mesh consists of 1308 elements being the subregion corresponding to soil A the biggest one with 784 elements while the pile is modelled with 144 elements.

Note that the block of a coefficient matrix corresponding only to soil A with 21168 dofs would require about 6.8 GB of memory while the maximum memory used employing the SCIA is 1.2 GB for the entire model which comprises 35316 dofs.

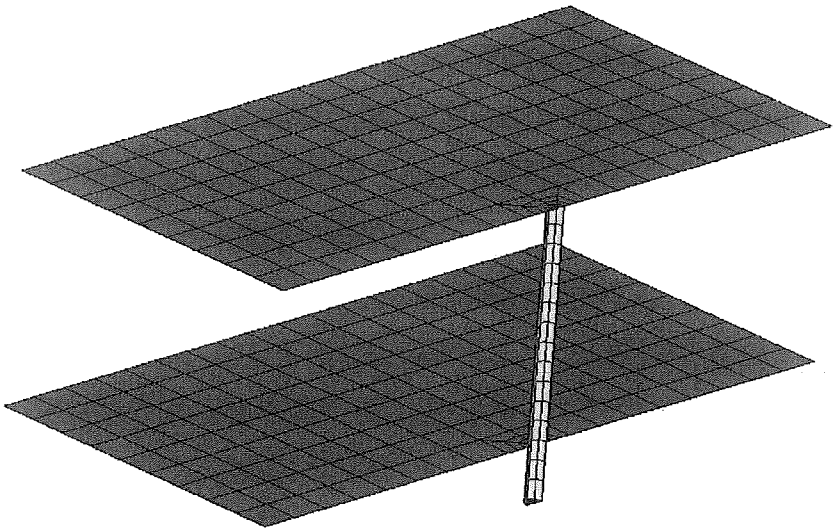


Figure 4: isometric view of half of the mesh used

Expressing the dynamic stiffness as follows

$$K_{ij} = K_{0ij}\left(k_{ij} + i\ a_0\ c_{ij}\right) \tag{9}$$

with K_{0ij} static stiffness

i imaginary unit

a_0 dimensionless frequency defined as $a_0 = \dfrac{2\ \omega\ r}{C_S}$

r pile radius (1 m)

ω excitation frequency (rad/s)

C_S shear wave velocity (of soil A)

k_{ij} stiffness coefficient

c_{ij} damping coefficient

one may identify the stiffness and damping coefficients pictured in Figure 5. The results obtained with the 3-d model, when compared to those obtained with the axisymmetric model, are satisfactory as only small differences between the curves are visible. Regarding the memory requirement the performance is quite satisfactory however, in terms of computation time there is still room for improvement, as several thousand iterations are necessary with the GMRES algorithm. It is thought that the computation times could be greatly reduced by implementing a preconditioner.

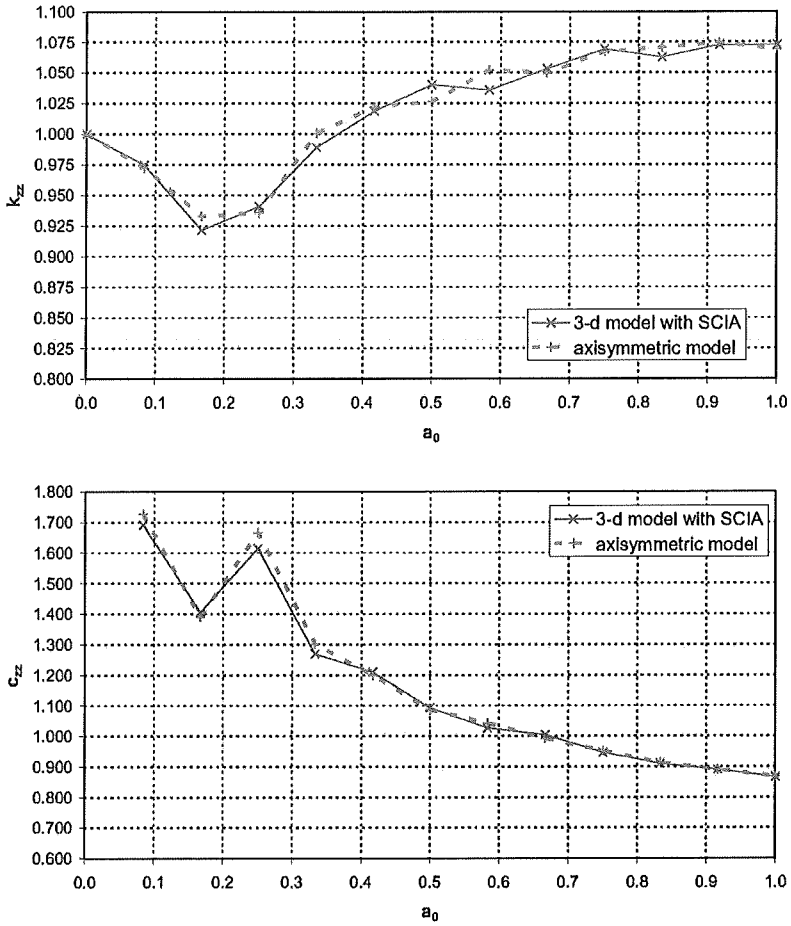


Figure 5: stiffness and damping coefficients for a single pile in layered soil

6 Conclusions

The present paper resumes the results obtained with a boundary element implementation combined with a similar coefficient identification algorithm (SCIA) that permits to treat problems with an ordinary personal computer, which normally would require very costly hardware equipment. The SCIA identifies pairs of collocation points and elements that present geometric similarities, which result in the same coefficients in the final system of equations. The gathered information permits to reduce the number of coefficients, named key coefficients, which have to be calculated by numerical integration while the remaining ones may be obtained by simple transformations. Apart from the reduced computation time for the integration, the key coefficients usually may be stored in the computer's RAM and thus be used to generate all remaining coefficients by simple transformations thereby permitting the matrix vector product in the GMRES algorithm to be

calculated without making use of a paging file which would be necessary when employing a conventional boundary element implementation.

The performance has been tested calculating the vertical dynamic stiffness of a pile foundation in layered soil. The results obtained are compared with those resulting from a axisymmetric formulation. The maximum amount of memory used during the solution process is about 1.2 GB for a model with more than 35000 dofs. This example indicates the potential of the combination of a conventional boundary element implementation with a similar coefficient identification algorithm.

References

[1] Brebbia, C.A. & Dominguez, J "Boundary Elements: An Introductory Course", McGraw-Hill Book Company, ISBN: 0-07-007414-3, 1989

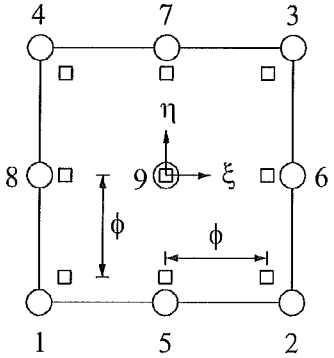
[2] Domínguez, J. "Boundary Elements in Dynamics", Elsevier Science, ISBN: 18-586-1021-4, 1993

[3] Beer, G. "Programming the Boundary Element Method", Wiley & Sons, ISBN 0-471-86333-5, 2001

[4] Guiggiani, M. & Gigante, A. "A general algorithm for multidimensional Cauchy principal value integrals in the boundary element method", ASME J. of Applied Mechanics , Vol. 57, pp. 906-915, 1990

[5] Saad, Y. & Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems", SIAM, Vol 7, 1986

Appendix



Node	ξ	η
1	-1	-1
2	1	-1
3	1	1
4	-1	1
5	0	-1
6	1	0
7	0	1
8	-1	0
9	0	0

Shape functions:

nodes 1-4: $N_i(\xi,\eta,\phi)=\frac{1}{4\phi^4}\xi(\xi+\xi_i\phi)\eta(\eta+\eta_i\phi)$

nodes 5-8: $N_i(\xi,\eta,\phi)=-\frac{1}{2\phi^4}\left(\xi_i^2\eta^2+\eta_i^2\xi^2-\phi^2\right)\left[\xi_i^2\xi(\xi+\xi_i\phi)+\eta_i^2\eta(\eta+\eta_i\phi)\right]$

node 9: $N_i(\xi,\eta,\phi)=\frac{1}{\phi^4}\left(\xi^2-\phi^2\right)\left(\eta^2-\phi^2\right)$

for geometrical nodes (circles) $\phi=1$, for support nodes (rectangles) $0<\phi<1$